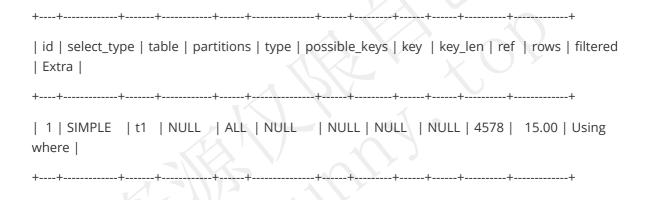
今天我们继续讲执行计划里的extra的信息,给大家讲一个平时最常见到的东西,就是**Using where**,这个恐怕是最最常见的了,其实这个一般是见于你直接针对一个表扫描,没用到索引,然后where里好几个条件,就会告诉你Using where,或者是你用了索引去查找,但是除了索引之外,还需要用其他的字段进行筛选,也会告诉你Using where。

比如说下面的SQL语句:

EXPLAIN SELECT * FROM t1 WHERE x2 = 'xxx'

这里的x2是没有建立索引的,所以此时他的执行计划就是下面这样的



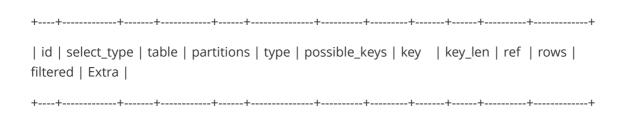
大家注意看,这里说了,针对t1表进行查询,用的是全表扫描方式,没有使用任何索引,然后全表扫描,扫出来的是4578条数据,这个时候大家注意看extra里显示了Using where,意思就是说,他对每条数据都用了WHERE x2 = 'xxx'去进行筛选。

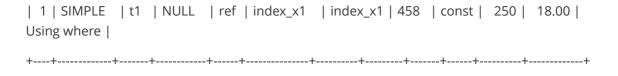
最终filtered告诉了你,过滤出来了15%的数据,大概就是说,从这个表里筛选出来了686条数据,就这个意思。

那么如果你的where条件里有一个条件是针对索引列查询的,有一个列是普通列的筛选,类似下面的 SQL语句:

EXPLAIN SELECT * FROM t1 WHERE x1 = 'xxx' AND x2 = 'xxx'

此时执行计划如下



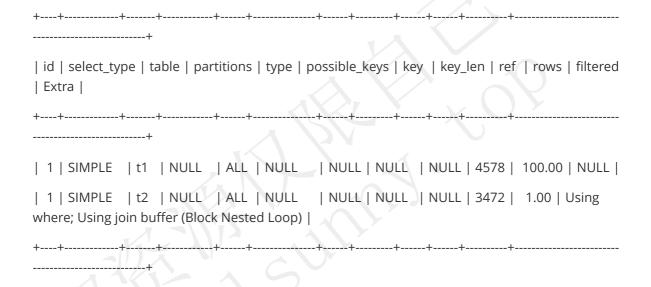


这个执行计划也是非常的清晰明了,这里针对t1表去查询,先通过ref方式直接在index_x1索引里查找,是跟const代表的常量值去查找,然后查出来250条数据,接着再用Using where代表的方式,去使用AND x2 = 'xxx'条件进行筛选,筛选后的数据比例是18%,最终所以查出来的数据大概应该是45条。

另外要给大家说的是,在多表关联的时候,有的时候你的关联条件并不是索引,此时就会用一种叫做join buffer的内存技术来提升关联的性能,比如下面的SQL语句:

EXPLAIN SELECT * FROM t1 INNER JOIN t2 ON t1.x2 = t2.x2

他们的连接条件x2是没有索引的,此时一起看看他的执行计划



这个执行计划其实也很有意思,因为要执行join,那么肯定是先得查询t1表的数据,此时是对t1表直接全表查询,查出来4578条数据,接着似乎很明确了,就是对每条数据的x2字段的值,跑到t2表里去查对应的数据,进行关联。

但是此时因为 t2 表也没法根据索引来查,也是属于全表扫描,所以每次都得对t2表全表扫描一下,根据extra提示的Using where,就是根据t1表每条数据的x2字段的值去t2表查找对应的数据了,然后此时会用join buffer技术,在内存里做一些特殊优化,减少t2表的全表扫描次数。

End